PATENT SPECIFICATION (11) 1281387

1281387

DRAWINGS ATTACHED

(21) Application No. 57241/69 (22) Filed 22 Nov. 1969

(45) Complete Specification published 12 July 1972

(51) International Classification G11C 15/00

(52) Index at acceptance

G4C 1F 2JX 2K 3F
G4A 10EX 18F 1F 2HX 6E
H3T 1M2P 2T3J 2T4 3X 4C 4M

(72) Inventor PETER LYCETT GARDNER

(54) ASSOCIATIVE STORE

(71) We, INTERNATIONAL BUSINESS MACHINES CORPORATION, a Corporation organized and existing under the Laws of the State of New York, United States of America, of Armonk, New York 10504, United States of America, do hereby declare the invention for which we pray that a patent may be granted to us, and the method by which it is to be performed, to be particularly described in and by the following statement:—

This invention relates to associative stores of the kind comprised of associative data storage cells or of the kind in which the associative interrogation function is performed externally of a storage array, thereby enabling non-associative data storage cells to be used in the array.

When an associative store is used to hold function tables for use in table-look-up operations, it has been found desirable that the storage cells of the memory be capable of assuming a "don't care" ("X") state for which a mismatch signal is not generated in response to interrogation for a binary one or a binary zero. By permitting the storage cells to ignore interrogation in this way the size of function tables is very much reduced.

Figure 1*a* shows, for example, a table of the function A *and* B for three-bit operands A, B respectively implemented in an associative store having two-state storage cells, the states representing binary one and binary zero respectively. If the A and B fields both match a search argument, a result field R containing the function A *and* B is read out. Since a cell cannot ignore an interrogation but must issue a mismatch signal if its contents do not match an interrogation the table must contain all possible combinations of the three bits which would give a non-zero result to the operation. This necessitates using thirty-seven lines of the store. Only thirteen are shown to indicate the pattern of the table. Using "don't care" state in the search field the table shown in Figure 1b occupies only three lines of the store, provided that facilities are available for multiple read-out of selected lines of the store with *or*-ing of the same orders of simultaneously accessed words. If, for example, operand A is 111 and operand B is 101, the first and third lines of the store are selected and the result fields 100 and 001 are read out simultaneously to give the resultant 101.

Heretofore it has been thought that the use of a "don't care" state in an associative store necessarily implies that the storage cells should have at least three stable states, as exemplified by the cells described in British Patent Specification 1,127,270. This is a great disadvantage since the complexity and cost of construction of a store employing three-state storage cells is greater than for a store of two-state cells.

According to the invention an associative store comprises a plurality of data storage cells which are capable in operation of assuming only two stable states, wherein one of the states is such that a mismatch signal is not generated if the data content of the cell is interrogated, irrespective of whether the cell is interrogated for a binary one or a binary zero.

The invention will be further explained, by way of example, with reference to the accompanying drawings, in which:—

FIGURES 1*a* and 1*b*, as already described, are function tables illustrating the use of the X state;

FIGURE 2 is a circuit diagram of a two-state data storage cell suitable for use in a store according to the invention:

FIGURES 3*a* and 3*b* are function tables;

FIGURE 4 is a block diagram of part of an associative store according to the invention;

FIGURE 5 is a block diagram of one column of another store according to the invention;

FIGURE 6 is a block diagram one column of another store according to the invention;

FIGURE 7 is a block diagram of one column of another store according to the invention;

FIGURE 8 is a block diagram of another store according to the invention;

FIGURE 9 is a circuit diagram of another two-state data storage cell useful for an associative store according to the invention;

FIGURE 10 is an arrangement of associative stores according to the invention for performing binary addition; and,

FIGURES 11 and 12 are function tables.

Figure 2 shows one example of a two state cell which can be used in an associative store according to the invention. The cell shown in Figure 2 is essentially one half of the four-state cell described in British Patent Specification 1,127,270 mentioned above. Transistor T1 is directly cross-coupled between collectors and bases with double-emitter transistor T2. Emitter E21 is directly connected to a word sense line 23 and emitter E22 is directly connected to a bit line 24. The cells of an associative store are arranged in rows and columns, all cells in the same row sharing a word sense line 23 and all cells in the same column sharing a bit line 24. The two states of the cell are the binary "1" state in which transistor T2 is conductive, and the "X" state in which transistor T1 is conductive. Interrogation for the binary zero state is effected by raising the potential of the bit line 24 and lowering the potential of the word sense line 23. If transistor T2 is conducting, indicating that the cell is in the binary one state, current will be directed through emitter E21 to appear on word sense line 23 as a mismatch signal. Interrogation for the binary one state is effected by lowering the potential on bit line 24. Whatever the state of the storage cell, no current will, as a result, be directed onto the word sense line 23 so that a mismatch will not be signalled. Clearly, if the cell is in the X state no current will appear on the word sense line whatever interrogation signal is applied on the bit line.

One form of associative store according to the invention is characterized by the feature that like orders of words selected by a search argument are *exclusive-or-ed* together on read out. How this can be done will be explained hereinafter, but in order to understand the concept involved reference should be made to the function tables shown in Figures 3*a* and 3*b*.

Figure 3*a* is a universal logic table for an associative store using three-state (1, 0, X) cells by means of which any of the sixteen possible logic functions of two variables, in this case four-bit operands, can be performed. The store is of a kind in which selected words are *or*-ed together on read out, i.e. the states of the cells of selected words are manifested simultaneously on the bit lines, and if not all the accessed cells of a given order stored binary zero or the X state, the result is a one bit in that order.

Blank spaces in Figures showing function tables represent cells which are in the X state. To use the table, a search argument comprising a key and the operands A and B is compared with the entries in the twelve-left-most columns of the tables. The result fields of those lines which match the contents of the search argument are read out simultaneously to give the result of the operation. The X state is read out as a binary zero. Alternatively, the result field could have been filled with binary zeroes but the table as shown in Figure 3*a* is a more direct comparison with the table to be discussed with reference to Figure 3b. Assume that, using the table of Figure 3*a*, an *exclusive-or* operation is to be performed on operand A, 1101, and operand B, 1010. The key is 0110 which selects the fifth to twelfth lines of the table. Mismatch of the operands with the contents of the fifth to twelfth columns of the table will cause rejection of lines five, six, eight, nine and eleven of the table leaving the arrowed lines selected. The result fields of these lines are XX1X, X1XX, and XXX1, which when read simultaneously give the result 0111. Tables constructed as in Figure 3*a* require, for an operation on two *n*-bit operands, 4*n* lines.

Figure 3*b* shows a universal logic table for an associative store using two-state (1, X) cells with the *exclusive-or-ing* together of different lines of data read from the store. The table is shown for four-bit operands. A table constructed as shown requires for an operation on two *n*-bit operands, $3n+1$ lines and is of the same width as a table of the kind shown in Figure 3*a*.

To do an *exclusive-or* operation the key is 0110, selecting the fifth to the twelfth rows of the table. If the operands are 1101 and 1010, as before, the five arrowed lines are selected and the result fields of these lines are *exclusive-or-ed* together on read out, thus $(1000)\forall(1011)\forall(0001)\forall(1000)\forall(0010)$, which gives the result 0111.

The last line of the table should especially be noted. By selecting this line a binary one is forced onto each but line of the result field, to produce as the final result the ones complement of whatever is the result of selecting other lines of the table. This may be readily understood if it is recalled that *exclusive-or-ing* a set of binary digits is the result of adding the digits (modulo 2). Adding a binary one to the sum (modulo 2) complements the sum.

Figure 4 shows one embodiment of an associative memory employing two-state storage cells 20 in which the words selected in an associative search are *exclusive-or-ed* together on readout. The cells 20 are arranged in rows and columns each row having a word sense line 23 and each column a bit line 24, connected to the cells as described with reference to Figure 2. Each bit line 24 is connected to the complement input 41 of a respective trigger 40. The triggers 40 are so

designed that a current pulse on the complement input changes the state of the trigger and thus complements the binary value represented by the state of the trigger. As pointed out above this is equivalent to doing an *exclusive-or* operation on successive signals appearing on a bit line 24. To read out, for example the result field of a table such as the universal logic table of Figure 3*b*, the potentials of the word sense lines 23 connected to storage cells which store the selected lines of the table are successively momentarily raised, causing the cells connected to the word sense lines to emit a current pulse on the bit line connected to a cell if the cell is in the binary one state. No pulse is emitted if the cell is in the X state. The current pulses appearing on a bit line 24 are accumulated, modulo 2, by the connected trigger 40. Before each read operation, the triggers 40 are set to the binary zero state. In order to energize the word sense lines successively the outputs of a distributor (not shown) are connected to the sense lines through gates (not shown) which are opened if the selector triggers of the rows of the associative store are set by an associative search operation.

The associative store shown in Figure 4, although economic in storage space (number of lines to a function table) is wasteful of the time, as the read cycle of the store is considerably lengthened. In the store part shown in Figure 5, the read cycle is short, since all lilnues are accessed simultaneously, but this gain is made at the expense of more components.

Figure 5 shows one column of an associative store with the references 20, 23 and 24 denoting the same elements as in Figure 4. The diagram is schematic and does not show the electronic circuit modifications, obvious to one skilled in the art, necessary to generate and to use in the manner to be explained, the outputs of the data cells 20. *Exclusive-or* circuits 51 to 53 are connected so that the outputs of circuits 51 and 52 provide respective inputs to circuit 53. The output of circuit 53 is connected to an input 54 of trigger 55. Input 54, when energized, sets trigger 55, which is initially in the binary zero state, to the binary one state. The outputs of the two top cells 20 of the column which are energized when the cells are being read out and are in the binary one state, are connected as respective inputs to *exclusive-or* circuit 51. The outputs of the two bottom cells of the column are connected as respective inputs to the *exclusive-or* circuit 52. It is clear that the input to trigger 55 is in the *exclusive-or* function of the contents of the column of the store. If a given row is not selected for read out, the contribution of that row to the inputs of the circuits 51 to 53 is a binary zero, which has no effect on the *exclusive-or* function of selected rows.

Figure 6 shows one column of another associative store according to the invention. For ease of reference the type of store shown in Figure 6 will be called a threshold store. The principle of operation of the threshold store is to detect the quantity of current emitted onto a bit line by the data storage cells being read onto the line. It can be assumed that the data storage cells are closely similar in their operating characteristics so that the quantity of current is directly proportional to the number of data cells in the binary one state. Accordingly, if the quantity of current is detectable, function tables can be constructed in which the number of binary ones in an order to the result field of selected words is significant.

In the threshold store of Figure 6, each bit line 24 is connected in parallel to sense amplifiers 61 and 62. The output of amplifier 61 and the inverted output of amplifier 62 are connected as respective inputs to *and* circuit 63, the output of which is connected to a trigger 64 so as to set the trigger to the binary one state when the output is energized. Amplifier 61 provides an enabling input to *and* circuit 63 when the quantity of current on bit line 24 is greater than or equal to the quantity emitted by one data cell 20. Amplifier 62 provides an output, which is inverted to disenable *and* circuit 63, when the quantity of current on bit line 24 is greater than or equal to twice the quantity emitted by one data cell 20. Put shortly, amplifier 61 produces a significant output if one or more binary ones are read onto bit line 24 and amplier 62 produces a significant output if two or more binary ones are read into bit line 24. The end result is that trigger 64, initially in the binary zero state, is set to the binary one state if and only if only one binary one is emitted onto the bit line 24.

The universal logic table for the threshold store of Figure 6 is as shown in Figure 3*b*. However not all the logic functions can be performed in one pass through the table. The effectiveness of the table depends on the fact that the result field of a line of the table can selectively be cancelled in accordance with the selection of other lines of the table. The criteria for cancellation in the case of the *exclusive-or* stores of Figures 4 and 5 are of broad application. If there is an odd number of ones simultaneously on a bit line, the result is one, and if there is an even number of ones on the bit the result is zero. This means that any number of sub-tables (the tables A, B, A.B, and complement of Figure 3*b*) can contribute to the generation of a result since the absolute number of ones on a bit line is not significant. However, in the threshold store the result is one if and only if only one one is on the bit line. Once two ones have been emitted onto the line, there is no way of changing the consequent result zero into a one. It follows that only

two sub-tables can be used in any operation and that the following five functions of the sixteen possible are not available in one table-look-up operation: A+B (key 1110), $\overline{A}$+B (key 1101), A+$\overline{B}$ (key 1011), A≡B (key 0111) and $\overline{A}$. $\overline{B}$ (key 1111). The keys listed are inadmissable in a threshold store and could be detected by an error table. The functions can be performed using the universal logic table by inverting one or both of the operands and then performing an operation which uses only two subtables. Thus, to perform A+B, both operands are inverted and

---

operation $\overline{A}$. $\overline{B}$ (key 1001) is effected. The other immediately unavailable operations are treated similarly.

It is useful to have a threshold store which has the possibility of simple *or-ed* read out or a different threshold. One column of such a store is shown in Figure 7. The arrangement of Figure 6 is modified by the provision of an *and-invert* circuit 71 having as one input the output of amplifier 62 and as the other input the 1 output of a trigger 72. The output of amplifier 62 is also connected as one input to an and circuit 73, the other input of which is the 0 output of a trigger 74. The 1 output of trigger 74 is connected as an input to *and* circuit 63, which also has as inputs the outputs of amplifier 61 and *and-invert* circuit 71. With both triggers 72 and 74 in the 0 state, the output of circuit 71 is always positive with the result that two inputs of *and* circuit 63 are always enabled. Trigger 64 is set to the 1 state whenever there is at least one one on bit line 24. With trigger 72 in the 1 state and tigger 74 in the 0 state, the arrangement of Figure 7 operates exactly as does the arrangement of Figure 6. With trigger 74 in the 1 state, irrespective of the state of trigger 72, *and* circuit 73 is enabled and *and* circuit 63 is disenabled. Trigger 64 is set to the 1 state if at least two ones are emitted simultaneously onto bit line 24.

Figure 8 shows another type of associative store according to the invention. The store, which uses two-state cells 20, is split into arrays 80A and 80B. The bit lines 24A, 24B of corresponding columns of each array are connected as respective inputs to an *exclusive-or* circuit 81, there being one such circuit for each pair of corresponding columns. The output of each circuit 81 is connected to a respective trigger 82 such that energization of the output sets the trigger to the binary one state.

The associative store of Figure 8 has all the desirable qualities of the *exclusive-or* store of Figure 4 but in addition each array has the *or* read out facility and can be accessed independently of, although not simultaneously with, the other array.

One drawback of the stores described with reference to Figures 4 to 8 is their unsuitability for general storage. Although, as has been briefly indicated, the stores offer certain advantages when used simply for table-look-up, two state cells as described with reference to Figure 2 do not lend themselves to the storage of binary data which is to searched for attributes of the data. This is especially so where the attributes are not under the designer's control as they are when designing a function table or the addresses of function tables. In the latter case, the designer may use, for example, an *n*-out-of-*m* code for addressing, instead of a pure binary code with X states. Thus, a field width of six 1—X cells will provide twenty independent addresses using a 3-out-of-6 code.

One way of overcoming the problem is to assign two columns of two state cells to each data bit over those fields where 1—0 cells are needed. The cell states are differentiated by the combinations of potentials on the two bit lines.

A more satisfactory approach is to construct the data storage cells as shown in Figure 9. Each cell 90 consists of two directly cross-coupled double-emitter transistors T3 and T4. Emitters E31 and E41 are respectively directly connected to bit lines 91 and 92. The remaining emitters E32 and E42 are directly connected to a word sense line 93. When used as a 1—X two-state cell, transistor T3 conducting represents the 1 state and transistor T4 conducting represents the X state. When a column of cells are assigned duty as 1—X cells, only bit line 91 is used for interrogation. The potential of bit line 92 is always kept low so that any current flowing in transistor T4 is drawn onto bit line 92 and cannot appear on word sense line 93. In this mode of operation the cell 90 simulates the 1—X cell 20. Alternatively, the cell 90 can be used as an ordinary two state associative cell in which binary zero is represented by transistor T4 being conductive. To interrogate for a binary one, the potential of bit line 92 is raised and that of bit line 91 is lowered. If the cell is in the one state, the current flowing in transistor T3 is steered onto bit line 91. If the cell is in the zero state, the potential on bit line 92 steers the current flowing in transistor T4 through emitter E42 to word sense line 93, thus generating a mismatch signal.

Assignment of a column of cells 90 to duty as a 1—X cell or a binary associative cell can be determined by a trigger which maintains low potential on bit line 92 if the trigger is in a given stable state. The triggers for all the columns of a store constitute a control register which can be loaded when the store is loaded or during execution of a program.

Addition is the operation by which the efficiency of a data processing system can most readily, albeit only approximately, be assessed. With an associative store in which the result fields are *or-ed* together on read out, an addition in one cycle of the store of two n bit operands takes $(10\times2^n-4n-9)$ lines of store. For $n=8$, 2,159 lines are needed. Clearly, this is not a desirable figure. By use of the threshold store of Figures 6 or 7 or the split array store of Figure 8 the number of lines required can be reduced to 53.

Binary addition can be defined in terms of the partial sum

$$A(m)\forall \qquad \text{and the carry}$$

$$A(m)\ .\ B(m)=G(m)$$

of order m of operands A and B. Order $(m+1)$ of the result R is: —

$$R(m+1)=P(m+1)\forall C(m)$$

where

$$C(m)=G(m)+P(m)\ .\ G(m-1)+P(m)\ .\ P(m-1)\ .\ G(m-2)+(P(m)\ .\ .\ .\ P(2)\ .\ P(1)\ .\ C(0))$$

and C(0) is the carry in to the lowest order (order 1).

Referring to Figure 10, associative stores 100 and 101 are either threshold stores of the kind shown in Figures 6 or 7 or split array stores as shown in Figure 8. Lines 102 and 103 of store 100 each contain a respective operand written twice, and a number of binary zeros. The arrangement is such that operands A and B occupy the same columns of the store and that operands A and B respectively occupy the same orders as the binary zeroes. If the store is the split array store of Figure 8, lines 102 and 103 are in different arrays. If the store is the threshold store of Figure 7, latches 72 and 74 are set so that the store operates in the same way as the store of Figure 6. A read operation on lines 102 and 103 leads to the generation of

$$A\forall B=P,\ A,\ \text{and}\ B,$$

as shown in Figure 10. The table 104 in store 101 generates from the inputs A and B the function $G=A\ .\ B$, and from the functions P and G the result of the addition.

Figure 11 shows the table 104 for the threshold store of Figures 6 or 7, and Figure 12 shows the table for the split-array store of Figure 8. The table is illustrated for four-bit operands and requires 19 lines as against 135 lines for an or read out store.

To illustrate how the tables are constructed, a short discussion of Figure 12 will be given The respective arrays A and B, which correspond to arrays 80A and 80B of Figure 8 each contain lines which define conditions for the existence of result one bits. But those conditions for the existence of result one bits. But those conditions which cannot coexist and give a result one bit are arranged in different arrays so that the simultaneous emission by each array of a one bit leads to the generation of a zero result bit. Consider, for example, the result bit R(4). This is one if

$$P(4)=A(4)\forall B(4)=1$$

or if there is a carry in from a lower order, but not if both conditions exist. The fourth line of array A gives $R(4)=1$ if $P(4)=1$, and the last four lines of array B give $R(4)=1$ if there is a carry in. If both arrays emit $R(4)=1$ then the output of the *exclusive-or* circuit 81 in the R(4) column of the result field leads to $R(4)=0$.

Although this invention has been described as embodied in an associative store with associative data storage cells, i.e. with cells which are constructed so as to respond to interrogation signals with match or mismatch output signals in accordance with the cell data content, it is equally applicable to associative stores using conventional binary cells, for example, with some modification of the logic circuitry, the associative store described in the specification of our copending application 34101/69 (Serial Number 1,220,000). It is consistent with the invention that interrogation of data can take place with the data in the cell or transferred to logic circuitry external to the storage array comprised of data cells.

WHAT WE CLAIM IS:—

1. An associative store comprising a plurality of data storage cells which are capable in operation of assuming only two stable states wherein one of the states (the X state) is such that a mismatch signal is not generated if the data content of the cell is interrogated, irrespective of whether the cell is interrogated for a binary one or a binary zero.

2. A store as claimed in claim 1, wherein each data storage cell comprises a single emitter transistor directly cross-coupled to a double-emiter transistor, one emitter of the double emitter transistor being connected to a word sense line and the X state being defined by the single emitter transistor being conductive.

3. A store as claimed in claim 1 or claim 2, wherein each data storage cell is constructed and arranged to operate substantially as described with reference to Figure 2.

4. A store as claimed in claim 1, wherein each data storage cell comprises two directly cross-coupled double-emitter transistors, one emitter of each transistor being connected to a respective bit line and the other emitter

of each transistor being directly connected to a word sense line.

5. A store as claimed in claim 4, wherein each data storage cell is constructed and arranged to operate substantially as described with reference to Figure 9 of the accompanying drawings.

6. A store as claimed in any one of the preceding claims wherein like orders of all words (A, B . . .) selected in a single associative search operation are *exclusive-or-ed* together (A$\forall$B$\forall$ . . .)or read out.

7. A store as claimed in claim 6, wherein the storage cells are arranged in rows and columns and each column of cells has a respective bit line which is connected to a complement input of a binary storage trigger, and to which signals representative of the states of selected storage cells forming the column are serially applied.

8. A store as claimed in claim 6 wherein, for all columns, the storage cells forming a column of the store are connected in pairs as the first-level inputs to a multilevel tree of *exclusive-or* circuits.

9. An associative store substantially as described with reference to Figure 4 or Figure 5 of the accompanying drawings.

10. A store as claimed in any one of claims 1 to 5 wherein the storage cells are arranged in rows and columns and each column of cells has a respective bit line, the the arrangement being such that, upon readout, data cells in a given state each emit onto the bit line a unit quantity of current, and wherein each bit line is associated with a storage trigger and a control circuit which is responsive to the quantity of current on the bit-line to set the trigger to a given stable state.

11. A store as claimed in claim 10, wherein the control circuit is responsive to the presence of one and only one unit of current on the bit line to set the trigger to the given stable state.

12. A store as claimed in claim 11, wherein the control circuit is responsive to the presence of two or more units of current on the bit line to set the trigger to the given stable state.

13. A store as claimed in claim 10, wherein the control circuit is selectively controllable to be responsive as claimed in claim 11, or as claimed in claim 12.

14. An associative store substantially as described with reference to Figure 6 or Figure 7 of the accompanying drawings.

15. A store as claimed in any one of claims 1 to 5, wherein the store comprises two identical like-ordered arrays of data storage cells and each column of each array has a respective bit line which is energized on read out in accordance with the data content of the cells in the column being read out, the bit lines of each corresponding order of the arrays being connected as respective inputs to an *exclusive-or* circuit, there being one such circuit to each order.

16. An associative store substantially as described with reference to Figure 8.

17. An arrangement of associative stores to perform the function of binary addition, constructed and arranged to operate substantially as described with reference to Figure 10.

LAWRENCE PERRY,
Chartered Patent Agent,
Agent for the Applicants.

Printed for Her Majesty's Stationery Office, by the Courier Press, Leamington Spa, 1972.
Published by The Patent Office, 25 Southampton Buildings, London, WC2A 1AY, from which copies may be obtained.

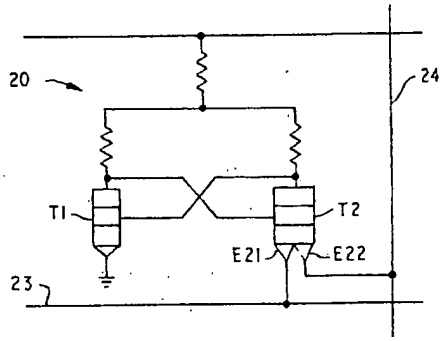
A			B			R		
1	1	1	1	1	1	1	1	1
1	1	1	1	1	0	1	1	0
1	1	1	1	0	1	1	0	1
1	1	1	1	0	0	1	0	0
1	1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1	0
1	1	1	0	1	1	0	1	1
1	1	0	1	1	1	1	0	0
1	1	0	1	1	0	1	1	0
1	1	0	1	0	1	1	0	0
1	1	0	1	0	0	1	0	0
1	1	0	0	1	0	0	1	0
1	1	0	0	1	1	0	1	0

FIG 1a

A			B			R		
1	X	X	1	X	X	1	0	0
X	1	X	X	1	X	0	1	0
X	X	1	X	X	1	0	0	1

FIG 1b

FIG. 2



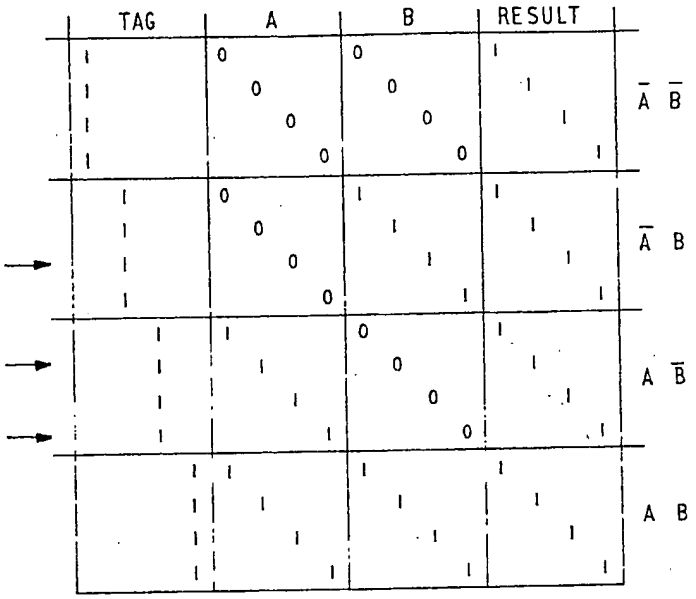

TAG
A
B
RESULT
$\bar{A}\ \bar{B}$
$\bar{A}$ B
A $\bar{B}$
A B

FIG 3a

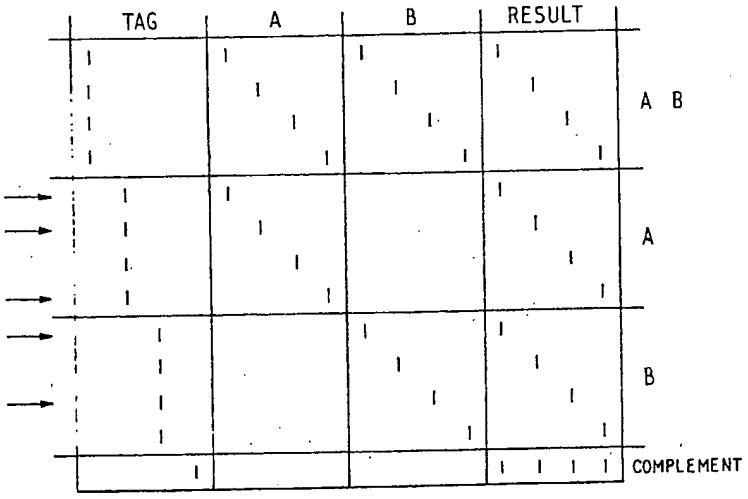

TAG
A
B
RESULT
A B
A
B
COMPLEMENT

FIG 3b

1281387 COMPLETE SPECIFICATION
8 SHEETS *This drawing is a reproduction of the Original on a reduced scale*
Sheet 3

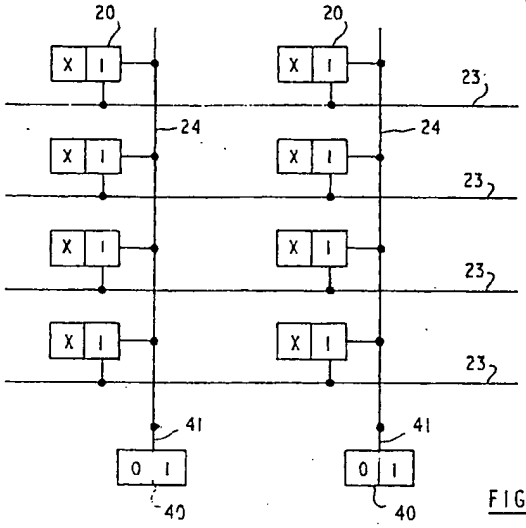


FIG 4

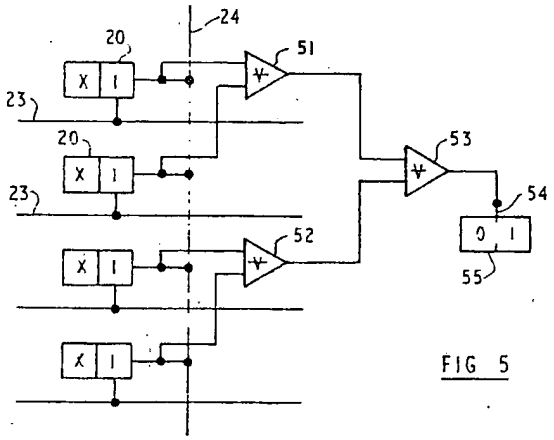


FIG 5

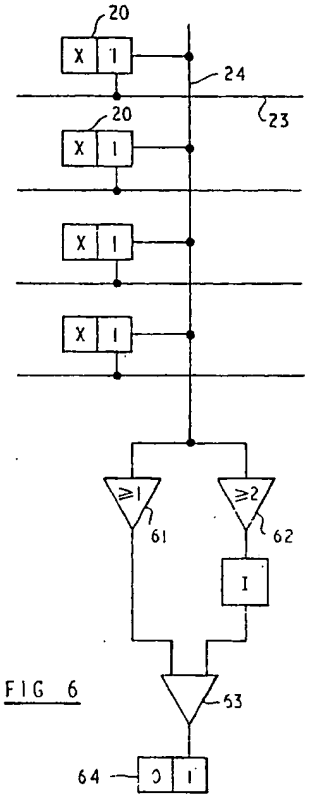

20
X
I
24
23
20
X
I
X
I
X
I
≥1
61
≥2
62
I
53
64
O
I

FIG 6

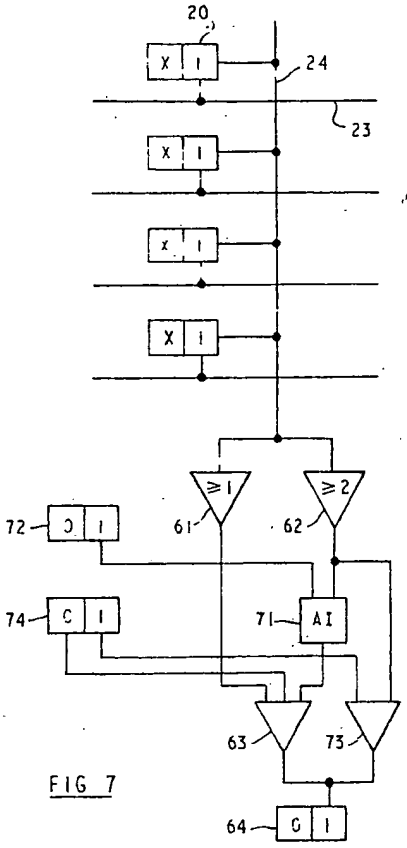

20
X
I
24
23
X
I
X
I
X
I
≥1
61
≥2
62
72
O
I
74
C
I
71
AI
63
73
64
O
I

FIG 7

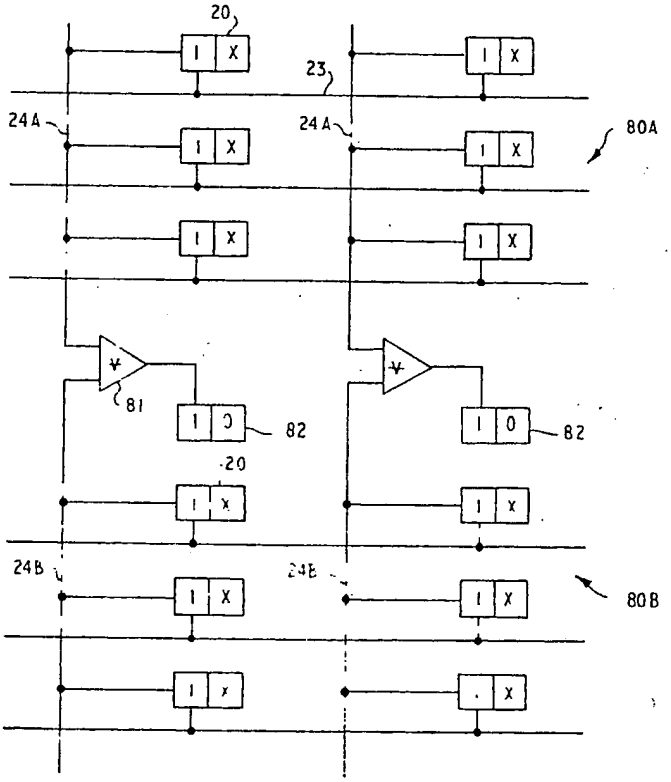


FIG 8

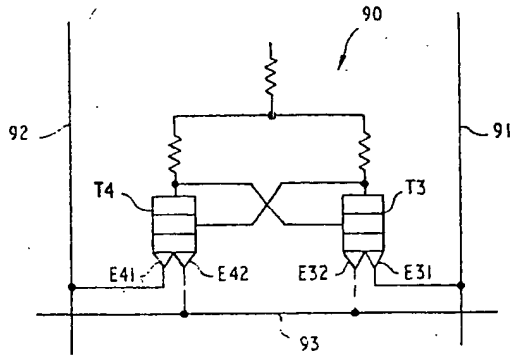

90
92
91
T4
T3
E41
E42
E32
E31
93

FIG 9

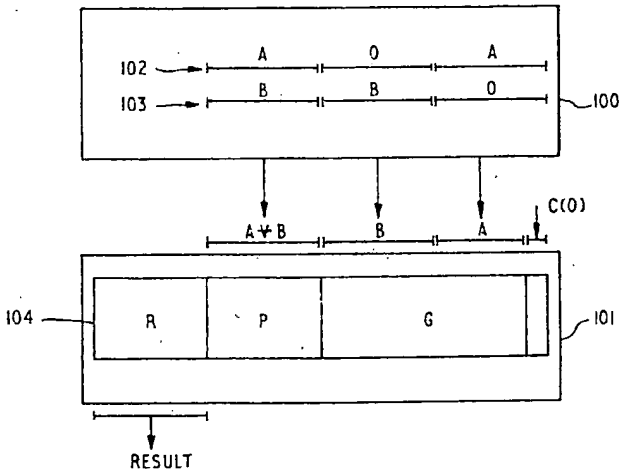

102
103
A
0
A
B
B
0
100
C(0)
A⊻B
B
A
104
R
P
G
101
RESULT

FIG 10

1281387 COMPLETE SPECIFICATION

8 SHEETS *This drawing is a reproduction of the Original on a reduced scale*

Sheet 7

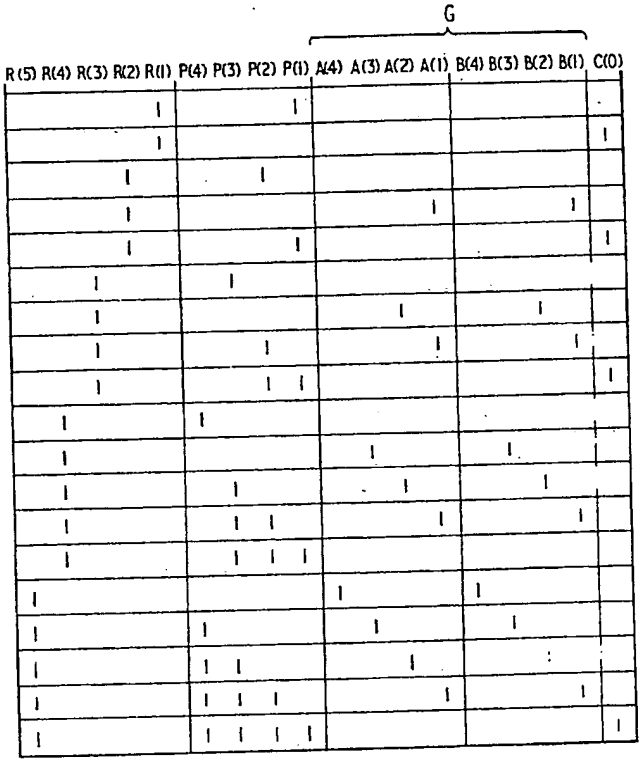


FIG 11

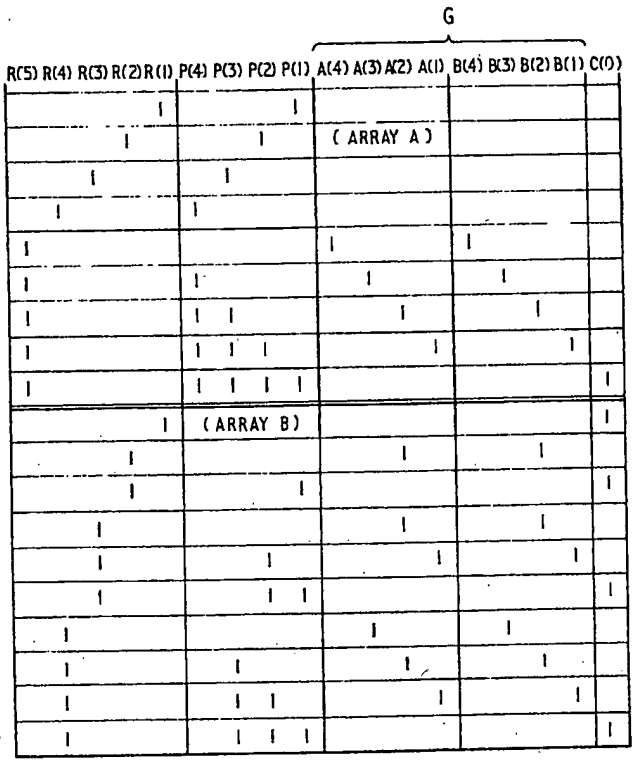


FIG. 12